LI 2022, CIRM (virtual) 1–2 February, 2022

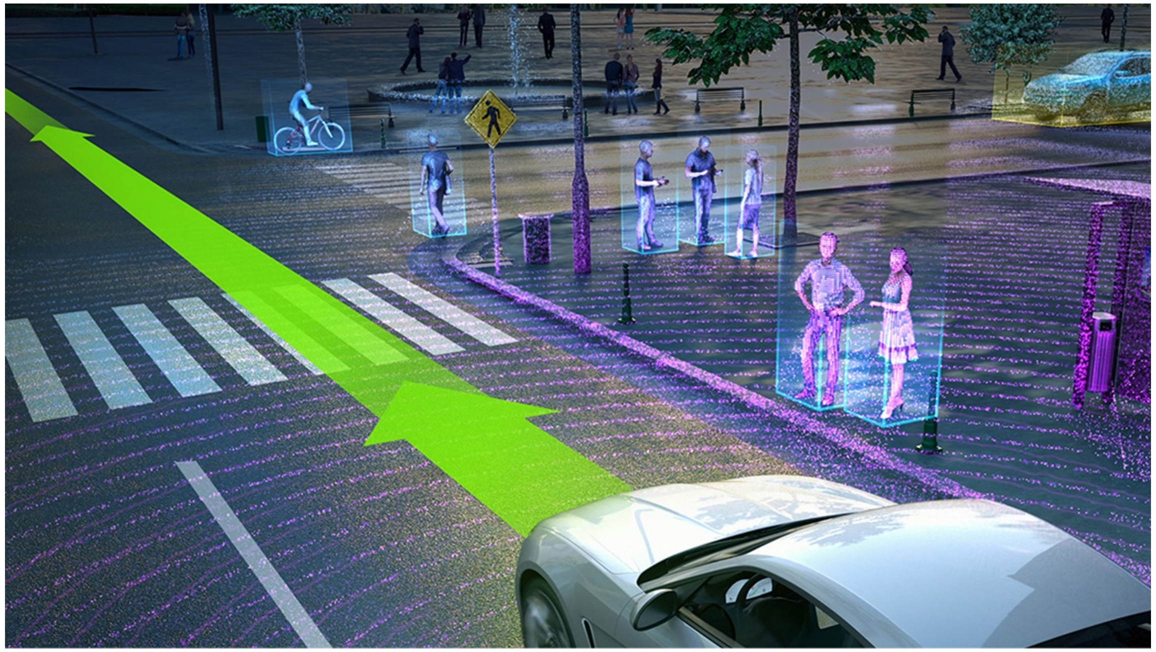
Probabilistic model checking for strategic equilibria-based decision making: Part 1

Prof. Marta Kwiatkowska

Department of Computer Science
University of Oxford

Software everywhere

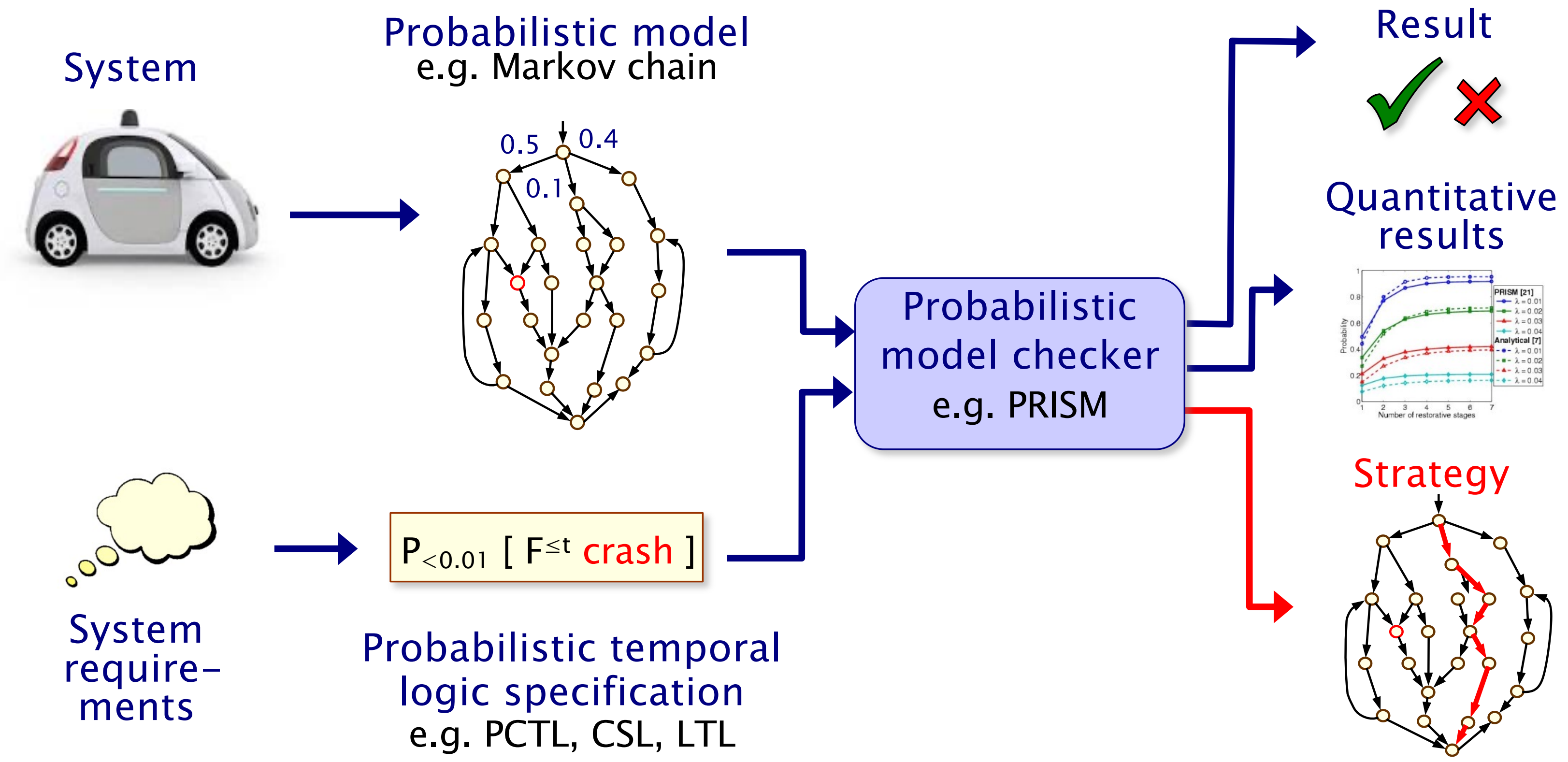
- Users expect: predictability & high integrity in presence of
  - component failure, environmental uncertainty, communication delays, …



- Safety, security, reliability, robustness, … can be expressed probabilistically
  - “the probability of an airbag failing to deploy within 0.02s is less than 0.001”

Probabilistic model checking

Automatic verification and strategy synthesis from temporal logic properties for probabilistic models

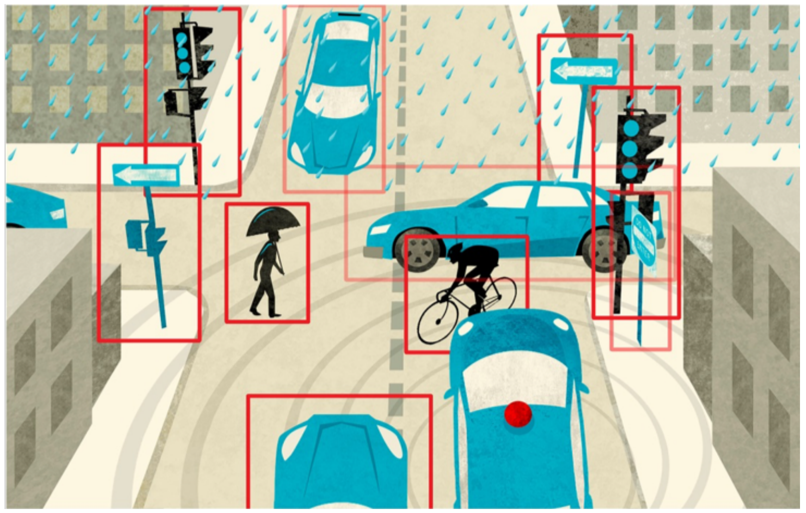


Tool support: PRISM

- First algorithms proposed in 1980s
  - algorithms [Vardi, Courcoubetis, Yannakakis, Hansson, Jonsson, de Alfaro…]
  - & first implementations
- 2001: general purpose tools released
  - PRISM: efficient extensions of symbolic model checking [Kwiatkowska, Norman, Parker, and many more …]
- Now mature area, of industrial relevance, new model checkers, tool competition
  - PRISM successfully used by non-experts in many domains
    - distributed algorithms, communication protocols, security protocols, biological systems, quantum cryptography, planning, robotics, …
  - genuine flaws found and corrected in real-world systems
  - www.prismmodelchecker.org

But which modelling abstraction?

- Complex decisions!
  - human and artificial agents
  - distinct goals
  - autonomy
  - competitive/collaborative behaviour
  - context
- Natural to adopt a game-theoretic view
  - need to account for the uncontrollable behaviour of agents or components, possibly with differing/opposing goals
  - in addition to controllable events
- Many occurrences in practice
  - e.g. decision making in economics, security, energy management, …



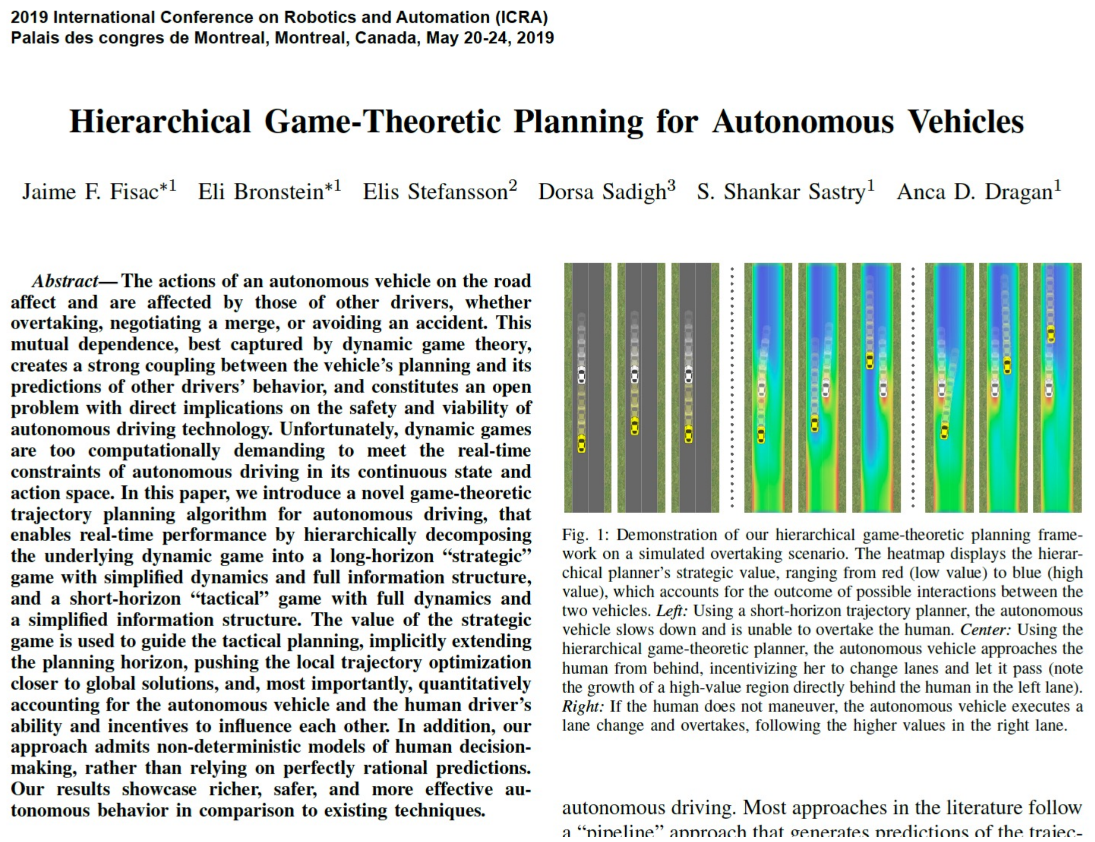
Why games?

- Games serve as abstractions for negotiation, strategic game playing and incentives to achieve more effective behaviour
  - virtual vs human agents, turn-based vs concurrent, zerosum vs non-zerosum, rational vs non-rational behaviours, individual vs social gain, …

- Relevant for a multitude of autonomous and AI scenarios
  - e.g. automated decisions, resource sharing, distributed coordination protocols, virtual assistants, etc
  - even autonomous driving…

Driving as a game-theoretic problem

- Merging into traffic difficult for autonomous cars
- Human drivers are not behaving rationally!
- Here dynamic games, and interplay between long-horizon strategic game playing with short-horizon tactical moves, plus incentives



2019 International Conference on Robotics and Automation (ICRA)
Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019

Hierarchical Game-Theoretic Planning for Autonomous Vehicles

Jaime F. Fisac*[1] Eli Bronstein*[1] Elis Stefansson[2] Dorsa Sadigh[3] S. Shankar Sastry[1] Anca D. Dragan[1]

***Abstract*— The actions of an autonomous vehicle on the road affect and are affected by those of other drivers, whether overtaking, negotiating a merge, or avoiding an accident. This mutual dependence, best captured by dynamic game theory, creates a strong coupling between the vehicle's planning and its predictions of other drivers' behavior, and constitutes an open problem with direct implications on the safety and viability of autonomous driving technology. Unfortunately, dynamic games are too computationally demanding to meet the real-time constraints of autonomous driving in its continuous state and action space. In this paper, we introduce a novel game-theoretic trajectory planning algorithm for autonomous driving, that enables real-time performance by hierarchically decomposing the underlying dynamic game into a long-horizon "strategic" game with simplified dynamics and full information structure, and a short-horizon "tactical" game with full dynamics and a simplified information structure. The value of the strategic game is used to guide the tactical planning, implicitly extending the planning horizon, pushing the local trajectory optimization closer to global solutions, and, most importantly, quantitatively accounting for the autonomous vehicle and the human driver's ability and incentives to influence each other. In addition, our approach admits non-deterministic models of human decision-making, rather than relying on perfectly rational predictions. Our results showcase richer, safer, and more effective autonomous behavior in comparison to existing techniques.**

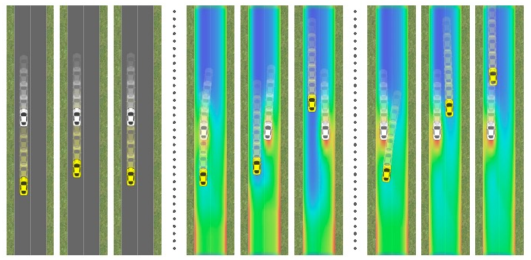


Fig. 1: Demonstration of our hierarchical game-theoretic planning framework on a simulated overtaking scenario. The heatmap displays the hierarchical planner's strategic value, ranging from red (low value) to blue (high value), which accounts for the outcome of possible interactions between the two vehicles. *Left:* Using a short-horizon trajectory planner, the autonomous vehicle slows down and is unable to overtake the human. *Center:* Using the hierarchical game-theoretic planner, the autonomous vehicle approaches the human from behind, incentivizing her to change lanes and let it pass (note the growth of a high-value region directly behind the human in the left lane). *Right:* If the human does not maneuver, the autonomous vehicle executes a lane change and overtakes, following the higher values in the right lane.

autonomous driving. Most approaches in the literature follow a "pipeline" approach that generates predictions of the trajec-

This mini-lecture course…

- Overview of stochastic multi-player games
  - modelling abstraction for competitive/cooperative behaviour, in adversarial environments
  - stochasticity to model e.g. failure, sensor uncertainty
  - turn-based and concurrent games
- Property specification: rPATL (based on PCTL and ATL)
  - zerosum and equilibria
  - model checking and strategy synthesis
  - case studies
- Tool support: PRISM-games 3.0
- Challenges and future directions

- Part 1 will cover zerosum games, Part 2 equilibria

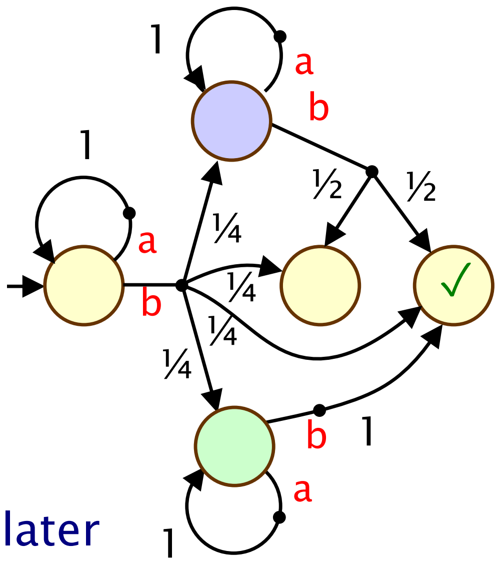
PRISM-games 3.0: Stochastic Game Verification with Concurrency, Equilibria and Time, Kwiatkowska et al., In *Proc* CAV 2020

Stochastic multi-player games (SMGs)

- A stochastic game involves
  - multiple players (competitive or collaborative behaviour)
  - nondeterminism (decisions, control, environment)
  - probability (failures, noisy sensors, randomisation)
- Game variants
  - turn-based vs concurrent, zero sum vs distinct goals
  - here, complete information games
- Widely studied, esp. algorithmic complexity, many applications
  - autonomous traffic (risk averse vs risk taking)
  - distributed coordination (selfish agents vs unselfish)
  - controller synthesis (system vs. environment)
  - security (defender vs. attacker)

Stochastic multi-player games

- Stochastic multi-player game (SMGs)
  - multiple players + nondeterminism + probability
  - generalisation of MDPs: each state controlled by unique player
- A (turn-based) SMG is a tuple $(\Pi, S, \langle S_i \rangle_{i \in \Pi}, A, \Delta, L)$:
  - Π is a set of n players
  - S is a (finite) set of states
  - $\langle S_i \rangle_{i \in \Pi}$ is a partition of S
  - A is a set of action labels
  - $\Delta : S \times A \rightarrow Dist(S)$ is a (partial) transition probability function
  - $L : S \rightarrow 2^{AP}$ is a labelling with atomic propositions from AP
- NB also called TSGs, concurrent games (CSGs) coming later



Rewards

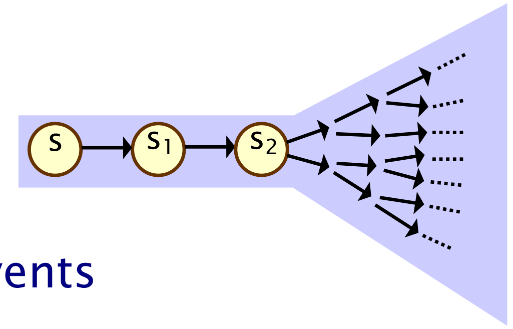
- Annotate SMGs with rewards (or costs)
  - real-valued quantities assigned to states and/or transitions
- Wide range of possible uses:
  - elapsed time, power consumption, number of messages successfully delivered, net profit, …
- We work with:
  - state rewards: $r : S \rightarrow \mathbb{R}_{\geq 0}$
  - action rewards: $r : A \rightarrow \mathbb{R}_{\geq 0}$
- Form basis for a variety of quantitative objectives
  - expected cumulative (total) reward (denoted C)
  - mean-payoff (limit-average) reward (denoted S)
  - ratio reward
  - (and many more not considered here)

Paths, strategies + probabilities

- A path is an (infinite) sequence of connected states in SMG
  - i.e. $s_0a_0s_1a_1\ldots$ such that $a_i \in A(s_i)$ and $\Delta(s_i,a_i)(s_{i+1})>0$ for all i
  - represents a system execution (i.e. one possible behaviour)
  - to reason formally, need a probability space over paths
- A strategy for player $i \in \Pi$ resolves choices in S_i states
  - based on history of execution so far
  - i.e. a function $\sigma_i : (SA)^*S_i \rightarrow \mathrm{Dist}(A)$
  - Σ_i denotes the set of all strategies for player i
  - deterministic if σ_i always gives a Dirac distribution
  - memoryless if $\sigma_i(s_0a_0\ldots s_k)$ depends only on s_k
  - also finite-memory, infinite memory, …
  - history based or explicit memory representation
- A strategy profile is tuple $\sigma=(\sigma_1,\ldots,\sigma_n)$
  - combining strategies for all n players

Paths, strategies + probabilities…

- For a strategy profile σ:
  - the game's behaviour is fully probabilistic
  - essentially an (infinite-state) Markov chain
  - yields a probability measure Pr_s^σ over set of all paths $Path_s$ from s



- Allows us to reason about the probability of events
  - under a specific strategy profile σ
  - e.g. any (ω-)regular property over states/actions

- Also allows us to define expectation of random variables
  - i.e. measurable functions $X : Path_s \rightarrow \mathbb{R}_{\geq 0}$
  - $E_s^\sigma [X] = \int_{Path_s} X \, dPr_s^\sigma$
  - used to define expected costs/rewards…

Property specification: rPATL

- Temporal logic rPATL:
  - reward probabilistic alternating temporal logic
- CTL, extended with:
  - coalition operator $\langle\langle C \rangle\rangle$ of ATL (Alternating Temporal Logic)
  - probabilistic operator P of PCTL, where $P_{\bowtie q}[\psi]$ means "the probability of ensuring ψ satisfies $\bowtie q$"
  - reward operator R of PRISM, where $R_{\bowtie q}[\rho]$ means "the expected value of ρ satisfies $\bowtie q$"
- Example:
  - $\langle\langle\{1,2\}\rangle\rangle\ P_{<0.01}[\ F^{\leq 10}\ \text{error}\ ]$
  - "players 1 and 2 have a strategy to ensure that the probability of an error occurring within 10 steps is less than 0.1, regardless of the strategies of other players"

rPATL properties

- Syntax (fragment):

  $\phi ::= \langle\langle C\rangle\rangle P_{\bowtie q}[\psi] \mid \langle\langle C\rangle\rangle R^{r}_{\bowtie q}[\rho] \mid \langle\langle C\rangle\rangle R^{r/c}_{\bowtie q}[\rho]$ ← "ratio"

  $\psi ::= F\ a$ ← "reachability"

  $\rho ::= C \mid S$ ← "cumulative", "longrun average"

- where:
  - $a \in AP$ is an atomic proposition, $C \subseteq \Pi$ is a coalition of players, $\bowtie \in \{\leq,<,>,\geq\}$, $q \in \mathbb{R}_{\geq 0}$, r and c are reward structures

- $\langle\langle C\rangle\rangle P_{\geq 1}[F \text{ “end”}]$
  - “players in coalition C have a collective strategy to ensure that the game reaches an “end”-state almost surely, regardless of the strategies of other players”

rPATL properties

- Syntax (fragment):

$$\phi ::= \langle\langle C\rangle\rangle P_{\bowtie q}[\psi] \mid \langle\langle C\rangle\rangle R^{r}_{\bowtie q}[\rho] \mid \langle\langle C\rangle\rangle R^{r/c}_{\bowtie q}[\rho]$$

"ratio"

$$\psi ::= F\ a$$

"reachability"

$$\rho ::= C \mid S$$

"longrun average"

"cumulative"

- $\langle\langle C\rangle\rangle R^{fuel}_{<q}\ [C]$
  - "players in coalition C have a strategy to ensure that the expected total fuel consumption is less than q, regardless of the strategies of other players"
- $\langle\langle C\rangle\rangle R^{fuel/time}_{\leq q}\ [S]$
  - "players in coalition C have a strategy to ensure that the expected longrun fuel consumption per time unit is at most q, regardless of the strategies of other players"

rPATL semantics

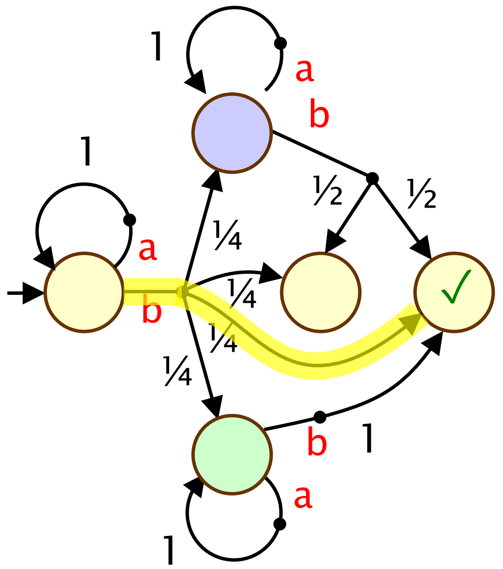
- Semantics for most operators is standard
- Just focus on P and R operators…
  - use reduction to a stochastic 2-player game

- Coalition game G_C for SMG G and coalition $C \subseteq \Pi$
  - 2-player SMG where C and $\Pi \setminus C$ collapse to players 1 and 2

- $\langle\langle C \rangle\rangle P_{\bowtie q}[\psi]$ is true in state s of G iff:
  - in coalition game G_C:
  - $\exists \sigma_1 \in \Sigma_1$ such that $\forall \sigma_2 \in \Sigma_2$. $Pr_s^{\sigma_1,\sigma_2}(\psi) \bowtie q$

- Semantics for R operator defined similarly…

Examples



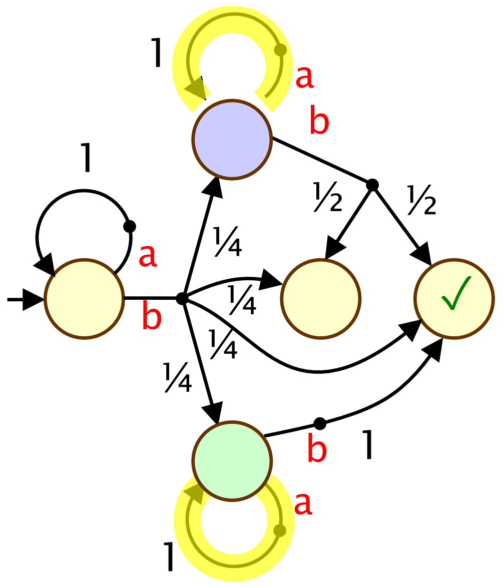
$\langle\langle$ 🟡 $\rangle\rangle P_{\geq 1/4}$[F ✓]

true in initial state

$\langle\langle$ 🟡 $\rangle\rangle P_{\geq 1/3}$ [F ✓]

$\langle\langle$ 🟡, 🟣 $\rangle\rangle P_{\geq 1/3}$ [F ✓]

Examples



⟨⟨🟡⟩⟩$P_{\geq 1/4}$[F ✓]

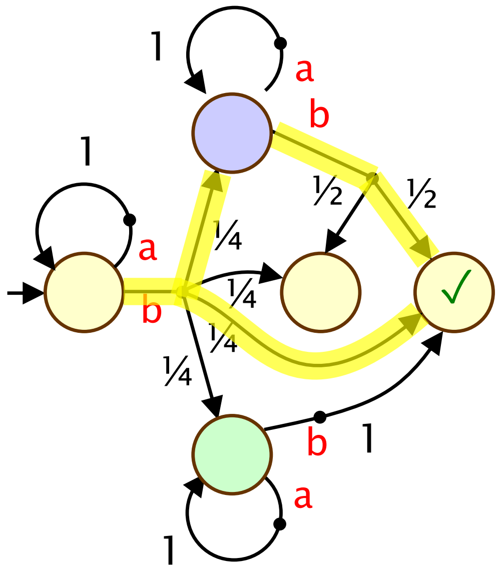
true in initial state

⟨⟨🟡⟩⟩$P_{\geq 1/3}$ [F ✓]

false in initial state

⟨⟨🟡,🔵⟩⟩$P_{\geq 1/3}$ [F ✓]

Examples



⟨⟨○⟩⟩$P_{\geq ¼}$[F ✓]

true in initial state

⟨⟨○⟩⟩$P_{\geq ⅓}$ [F ✓]

false in initial state

⟨⟨○,●⟩⟩$P_{\geq ⅓}$ [F ✓]

true in initial state

Verification and strategy synthesis

- The verification problem is:
  - Given a game G and rPATL property ϕ, does G satisfy ϕ?
- e.g. $\langle\langle C\rangle\rangle P_{\bowtie q}[\psi]$ is true in state s of G iff:
  - in coalition game G_C:
  - $\exists\sigma_1\in\Sigma_1$ such that $\forall\sigma_2\in\Sigma_2$. $Pr_s^{\sigma_1,\sigma_2}(\psi) \bowtie q$

- The synthesis problem is:
  - Given a game G and a coalition property ϕ, find, if it exists, a coalition strategy σ that is a witness to G satisfying ϕ

- Reduce to computing optimal values and winning strategies in 2-player games
  - e.g. $\langle\langle C\rangle\rangle P_{\geq q}[\psi] \Leftrightarrow \sup_{\sigma_1\in\Sigma_1} \inf_{\sigma_2\in\Sigma_2} Pr_s^{\sigma_1,\sigma_2}(\psi) \geq q$
  - complexity NP $\cap$ coNP (this fragment), cf P for MDPs

Automatic Verification of Competitive Stochastic Systems, Chen et al., In *Proc* TACAS 2012

Verification and strategy synthesis

- The verification problem is:
  - Given a game G and rPATL property ϕ, does G satisfy ϕ?
- The synthesis problem is:
  - Given a game G and a coalition property ϕ, find, if it exists, a coalition strategy σ that is a witness to G satisfying ϕ
- Reduce to computing optimal values and winning strategies in 2-player games
  - typically employ value iteration to specified convergence
  - both players have optimal strategies
  - memoryless deterministic strategies suffice
  - (epsilon-optimal) strategies can be typically extracted from optimal values in linear time

Example: Probabilistic reachability

- E.g. $\langle\langle C\rangle\rangle P_{\geq q}[\ F\ \phi\ ]$: max/min reachability probabilities
  - compute $\sup_{\sigma_1 \in \Sigma_1} \inf_{\sigma_2 \in \Sigma_2} Pr_s^{\sigma_1,\sigma_2}(F\ \phi)$ for all states s
  - deterministic memoryless strategies suffice

- Value $p(s)$ for state s is least fixed point of:

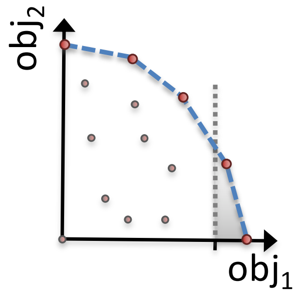
$$p(s) = \begin{cases} 1 & \text{if } s \in Sat(\phi) \\ \max_{a \in A(s)} \Sigma_{s' \in S}\ \delta(s,a)(s') \cdot p(s') & \text{if } s \in S_1 \backslash Sat(\phi) \\ \min_{a \in A(s)} \Sigma_{s' \in S}\ \delta(s,a)(s') \cdot p(s') & \text{if } s \in S_2 \backslash Sat(\phi) \end{cases}$$

- Computation (value iteration):
  - start from zero, propagate probabilities backwards
  - guaranteed convergence wrt “usual” termination criteria

Multi-objective properties

- May need to explore trade-offs
  - e.g. between performance and resource usage: maximise probability of success and minimise energy usage
- Consider conjunctions of objectives (for stopping games), also known as multidimensional
  - expected total rewards, mean-payoffs or ratios
  - almost sure mean-payoffs/ratios
- Example
  - "the expected longrun average fuel consumption and profit are simultaneously at least v1 and v2, respectively "

    $\langle\langle C\rangle\rangle\ (\ R^{fuel}_{\geq v1}\ [S]\ \&\ R^{profit}_{\geq v2}\ [S]\ )$



- NB Boolean combinations may be needed for implication

  $\langle\langle C\rangle\rangle\ (\ R^{fuel/time}_{\geq v1}\ [S] \Rightarrow R^{profit}_{\geq v2}\ [S]\ )$

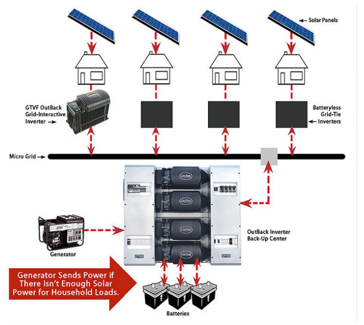
Compositional Strategy Synthesis for Stochastic Games with Multiple Objectives, Basset et al., Info&Comp 2018 27

Multi-objective properties

- For MDPs, optimal strategies exist but randomised strategies may be needed
- For stochastic games:
  - optimal strategies may not exist
  - infinite memory may be required
- Therefore
  - work with restricted games (e.g. stopping)
  - use stochastic memory update representation [Brazdil et al, 2014]
    - exponentially more succinct than deterministic update
    - equivalent power if infinite memory allowed
- Decision procedure
  - complexity is NP $\cap$ coNP
  - compute epsilon-approximations of Pareto sets and epsilon-optimal strategies, fixed point reached in finitely many steps

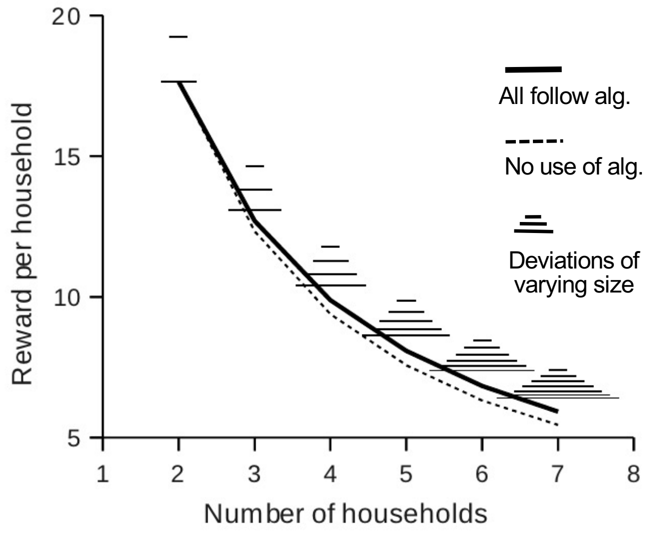
Case study: Energy management

- Energy management protocol for Microgrid
  - Microgrid: local energy management
  - randomised demand management protocol [Hildmann/Saffre'11]
  - probability: randomisation, demand model, …


Solar Panels
GTVF OutBack Grid-Interactive Inverter
Batteryless Grid-Tie Inverters
Micro Grid
OutBack Inverter Back-Up Center
Generator
Generator Sends Power if There Isn't Enough Solar Power for Household Loads.
Batteries

- Existing analysis
  - simulation-based
  - assumes all clients are unselfish

- Our analysis
  - stochastic multi-player game
  - clients can cheat (and cooperate)
  - exposes protocol weakness
  - propose/verify simple fix

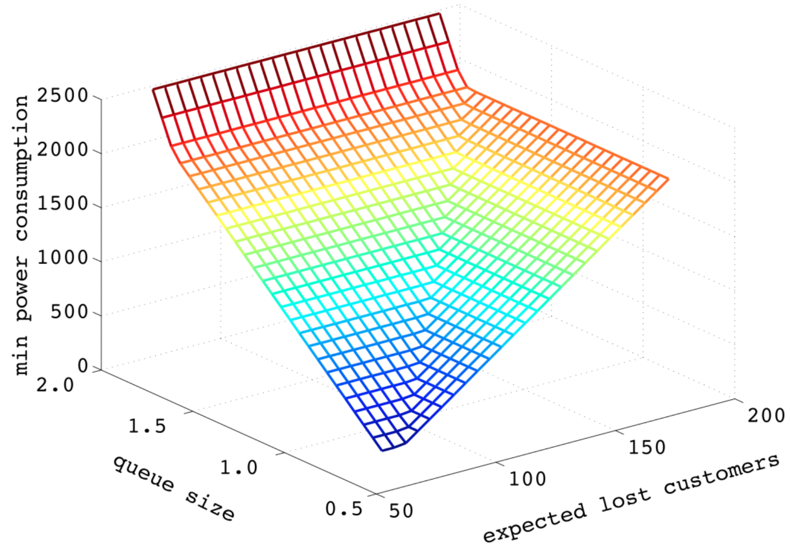


Automatic Verification of Competitive Stochastic Systems, Chen et al., In *Proc* TACAS 2012

Case study: Dynamic power management

- Synthesis of dynamic power management schemes
  - for an IBM TravelStar VP disk drive
  - 5 different power modes: active, idle, idlelp, stby, sleep
  - power manager controller bases decisions on current power mode, disk request queue, etc.

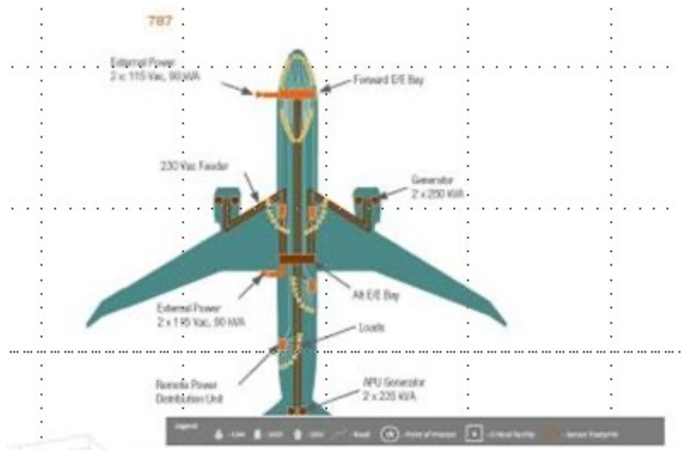
- Build controllers that
  - minimise energy consumption, subject to constraints on e.g.
  - probability that a request waits more than K steps
  - expected number of lost disk requests



Quantitative Multi-Objective Verification for Probabilistic Systems. Forejt et al, In *(TACAS'11)*, volume 6605 of LNCS, pages 112-127, Springer. March 2011.

Case study: Aircraft power distribution

- Consider Honeywell high-voltage AC (HVAC) subsystem
  - power routed from generators to buses through switches
  - represent as a stochastic game, modelling competition for buses, with stochasticity used to model failures
  - specify control objectives in LTL using longrun average
  - e.g. “maximise uptime of the buses and minimise failure rate”

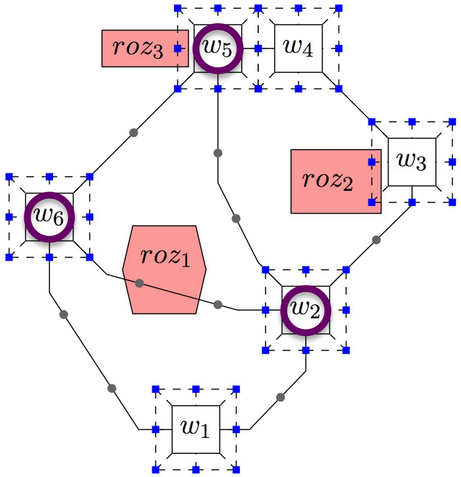


- Solution (PRISM-games 2.0)
  - compositional strategy synthesis
  - enable the exploration of trade-offs between uptime of buses and failure rate

Compositional Controller Synthesis for Stochastic Games, Basset et al., In *Proc* CONCUR 2014

Case study: UAV path planning

- Human operator
  - sensor tasks
  - high-level commands for piloting
- UAV autonomy
  - low-level piloting function
- Quantitative mission objectives
  - road network surveillance with the minimal time, fuel, or restricted operating zone visits
- Analysis of trade-offs
  - consider operator fatigue and workload
  - multi-objective, MDP and SMG models



Controller Synthesis for Autonomous Systems Interacting with Human Operators. L. Feng et al, In *Proc*. ICCPS 2015, ACM

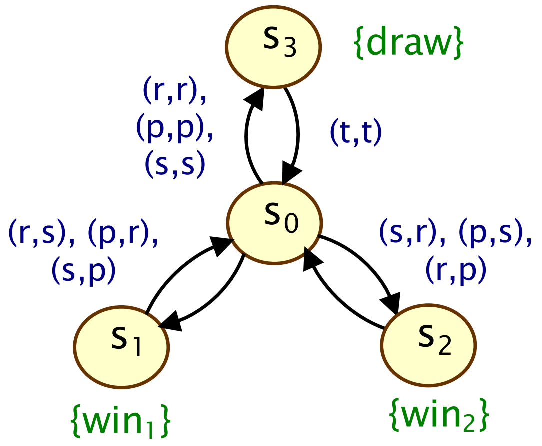
Concurrent stochastic games

- Concurrent stochastic games (CSGs)
  - players choose actions concurrently
  - jointly determines (probabilistic) successor state
  - generalises turn-based stochastic games
- Key motivation:
  - more realistic model of components operating concurrently, making action choices without knowledge of others
- Formally
  - set of n players N, state space S, actions A_i for player i
  - transition probability function $\delta : S \times A \rightarrow Dist(S)$
  - where $A = (A_1 \cup \{\perp\}) \times \ldots \times (A_n \cup \{\perp\})$
  - strategies $\sigma_i : FPath \rightarrow Dist(A_i)$, strategy profiles $\sigma=(\sigma_1,\ldots,\sigma_n)$
  - probability measure Pr_s^{σ}, expectations $E_s^{\sigma}(X)$

Example CSG: rock-paper-scissors

- Rock-paper-scissors game
  - 2 players repeatedly draw rock (r), paper (p), scissors (s), then restart the game (t)
  - rock > scissors, paper > rock, scissors > paper,
  - otherwise draw

- Example CSG
  - 2 players: N={1,2}
  - $A_1 = A_2 = \{r,p,s,t\}$
  - NB: no probabilities here



Matrix games

- Matrix games
  - finite, one-shot, 2-player, zero-sum games
  - utility function $u_i : A_1 \times A_2 \rightarrow \mathbb{R}$ for each player i
  - represented by matrix Z where $z_{ij} = u_1(a_i,b_j) = -u_2(a_i,b_j)$
- Example:
  - one round of rock-paper-scissors

$$Z = \begin{array}{c|ccc} & r & p & s \\ \hline r & 0 & -1 & 1 \\ p & 1 & 0 & -1 \\ s & -1 & 1 & 0 \end{array}$$

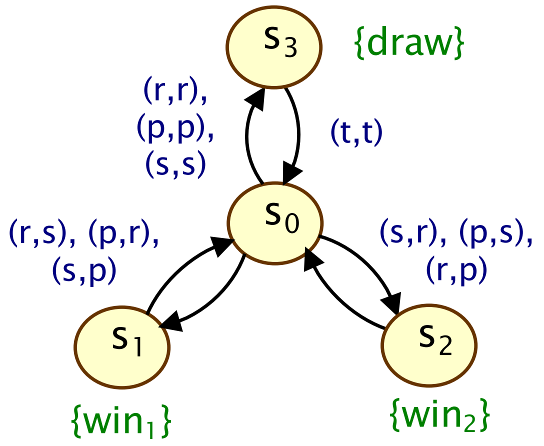
- Optimal (player 1) strategy via LP solution (minimax):
  - compute value val(Z): maximise value v subject to:
  - $v \leq x_p - x_s$
    $v \leq x_s - x_r$,
    $v \leq x_s - x_p$
    $x_r + x_p + x_s = 1$
    $x_r \geq 0$, $x_p \geq 0$, $x_s \geq 0$

Optimal strategy (randomised):
$(x_r,x_p,x_s) = (⅓,⅓,⅓)$

rPATL for CSGs

- We use the same logic rPATL as for SMGs

- Examples for rock-paper-scissors game:
  - $\langle\langle 1\rangle\rangle\ P_{\geq 1}$ [F win_1] – player 1 can ensure it eventually wins a round of the game with probability 1
  - $\langle\langle 2\rangle\rangle\ P_{max=?}$ [$\neg win_1$ U win_2] – the maximum probability with which player 2 can ensure it wins before player 1
  - $\langle\langle 1\rangle\rangle\ R_{max=?}$ [$C^{\leq 2K}$]^{utility_1}– the maximum expected utility player 1 can ensure over K rounds (utility = 1/0/−1 for win/draw/lose)



rPATL model checking for CSGs

- Extends model checking algorithm for SMGs
  - key ingredients are solution of (zero-sum) 2-player CSGs
- E.g. $\langle\langle C \rangle\rangle P_{\geq q}[\ F\ \phi\ ]$: max/min reachability probabilities
  - compute $\sup_{\sigma_1 \in \Sigma_1} \inf_{\sigma_2 \in \Sigma_2} Pr_s^{\sigma_1,\sigma_2}(F\ \phi)$ for all states s
  - note that optimal strategies are now randomised
  - solution of the 2-player CSG is in PSPACE
  - we use a value iteration approach
- Value $p(s)$ for state s is least fixed point of:
  - $p(s) = 1$ if $s \in Sat(\phi)$ and otherwise $p(s) = val(Z)$ where:
  - Z is the matrix game with $z_{ij} = \Sigma_{s' \in S}\ \delta(s,(a_i,b_j))(s') \cdot p(s')$
  - so each iteration requires solution of a matrix game for each state (LP problem of size $|A|$, where A = action set)

Automatic Verification of Concurrent Stochastic Games, Kwiatkowska et al., FMSD 2021

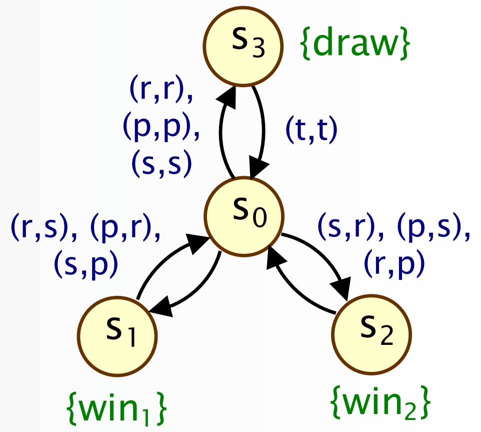
CSGs in PRISM-games

- CSG model checking implemented in PRISM-games
- Extension of PRISM modelling language
  - player specification via partition of modules
  - unlike SMGs, all modules move simultaneously
  - concurrent updates modelled with multi-action commands, e.g. [r1,r2] m1=0 → … and chained updates, e.g. (m2'=m1')
- Explicit engine implementation
  - use off-the-shelf LP solver for minimax LP solution
  - apply precomputation algorithms to filter out trivial states, remove states with infinite rewards
  - experiments with CSGs up to ~3 million states

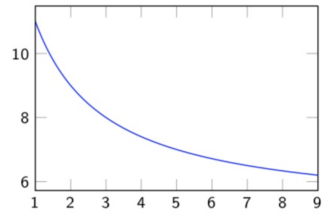
PRISM-games 3.0: Stochastic Game Verification with Concurrency, Equilibria and Time, Kwiatkowska et al., In *Proc* CAV 2020

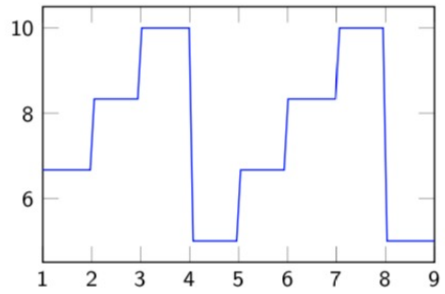
CSGs in PRISM (rock-paper-scissors)

```
csg

player player1 M1 endplayer
player player2 M2 endplayer

module M1
        m1 : [0..3];
        [r1] m1=0 → (m1'=1); // rock
        [p1] m1=0 → (m1'=2); // paper
        [s1] m1=0 → (m1'=3); // scissors
        [t1] m1>0 → (m1'=0); // restart
endmodule
module M2 = M1 [ m1=m2, r1=r2 , p1=p2, s1=s2, t1=t2 ]
endmodule

label "win1" = (m1=1&m2=3) | (m1=2&m2=1) | (m1=3&m2=2); //
player 1 wins round
rewards "utility1" // utility for player 1
        [t1] (m1=1 & m2=3) | (m1=2 & m2=1) | (m1=3 & m2=2) : 1;
// player 1 wins
        [t1] (m1=1 & m2=2) | (m1=2 & m2=3) | (m1=3 & m2=1) : -1;
// player 2 wins
endrewards
```



Case study: Future markets investor



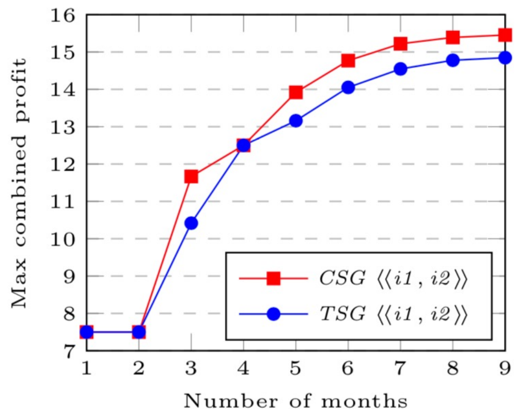


- Model of interactions between:
  - stock market, evolves stochastically
  - two investors i_1, i_2 decide when to invest
  - market decides whether to bar investors
- Modelled as a 3-player CSG
  - extends simpler model originally from [McIver/Morgan'07]
  - investing/barring decisions are simultaneous
  - profit reduced for simultaneous investments
  - market cannot observe investors' decisions
- Analysed with rPATL model checking & strategy synthesis
  - distinct profit models considered: 'normal market', 'later cash-ins' and 'later cash-ins with fluctuation'
  - comparison between turn-based and concurrent game models

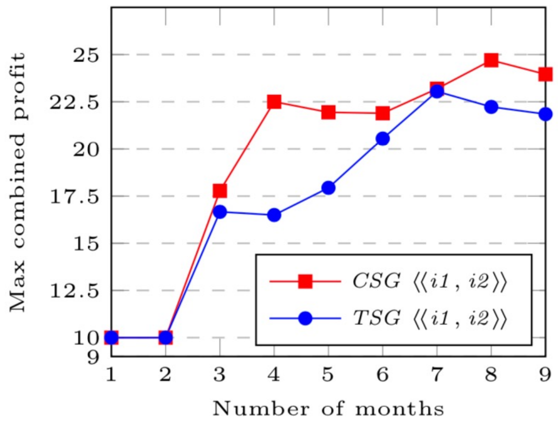
Case study: Future markets investor

- Example rPATL queries:
  - $\langle\langle \text{investor}_1 \rangle\rangle\ R^{\text{profit}_1}_{\max=?}\ [\ F\ \text{finished}_1\ ]$
  - $\langle\langle \text{investor}_1,\text{investor}_2 \rangle\rangle\ R^{\text{profit}_{1,2}}_{\max=?}\ [\ F\ \text{finished}_{1,2}\ ]$
  - i.e. maximising individual/joint profit

- Results (joint profit) – cooperation pays off in a concurrent game model
  - optimal (randomised) investment strategies synthesised



with fluctuations



without

Case studies (selected)

- Turn-based games
  - futures market investor model [McIver & Morgan]
  - energy management in microgrids [TACAS'12]
  - DNS bandwidth amplification attack [Deshpande et al]
  - self-adaptive software architectures [Camara, Garlan et al]
  - attack-defence scenarios in RFID goods managament [Aslanyan et al]
- Multi-objective turn-based games
  - UAV path planning with operator (multi-objective) [ICCPS'15]
  - aircraft electric power control (compositional) [TACAS'15]
- Concurrent games
  - public good game [CAV 2020]
  - intrusion detection policies [QEST 2018]
  - Aloha protocol [QEST 2020]

https://www.prismmodelchecker.org/games/casestudies.php

Summary Part 1

- Overview of stochastic multiplayer games
  - turn-based and concurrent
  - zerosum properties
  - logic rPATL (probability, rewards, coalitions)
  - model checking algorithms for zerosum
  - strategy synthesis

- Covered theory and tool implementation
  - PRISM-games 3.0
  - combination of value iteration, matrix games and LP solving
  - wide variety of case studies analysed

- Part 2: multiple objectives, hence equilibria

Acknowledgements

- My group and collaborators in this work
- Project funding
  - ERC Advanced Grant VERIWARE
  - EPSRC Mobile Autonomy Programme Grant

- See also
  - PRISM www.prismmodelchecker.org

- ERC Advanced Grant fun2model

  "From FUNction-based TO MOdel-based automated probabilistic reasoning for DEep Learning"